EECS 4415 Project 1

Hongru Wang
Department. of Electrical Engineering & Computer Science
York University, Toronto, Canada
bruce123@my.yorku.ca

Abstract -----Nowadays, finding sets of items that appear together from commercial dataset and mining association rules from the frequent itemset is a popular task with big data system. The hardest problem in the task mentioned above is finding frequent pairs. In this project, we are going to implement and analyze different algorithms for finding frequent pairs.

I. WEEK 1: APRIORI ALGORITHM

A. Introduction of Apriori Algorithm

Apriori algorithm uses monocity to erase candidate sets for counting. It is a 2-pass algorithm for finding frequent pairs.

Pass 1: Read baskets and count in main memory the occurrences of each individual item, and store frequent items whose occurrence is bigger than threshold

Pass 2: Read baskets again and count only those pairs where both elements are frequent in memory

B. Implementation of Apriori Algorithm

I implemented the Apriori Algorithm in the following steps:

1. Read dataset from the file
2. Generate the min_support threshold
3. Count the number of occurrences for each item in the dataset. This is for finding C1.
4. Filter items that occurrences are below threshold and store the frequent items. This is for finding L1
5. Use the frequent items to prune the dataset, if a basket does not contain any frequent item, then remove that basket
6. Use the frequent items to generate all unique pairs of 2 frequent items. This is for finding C2.
7. Count the number of occurrences of each pair and store the pair and its related number of occurrences in hash table
8. Filter pairs that occurrences are below threshold and store the frequent combinations. This is for finding L2

C. Result of Apriori Alforithm

1.The last column of the table represents the number of frequent pairs

2.The Netflix dataset runs out of time so there is no table built based on the Netflix dataset

Retail dataset

Time (s)	Threshold (%)	Frequent pairs
115.18 s	1%	58
9.83 s	2%	22

Netflix dataset: runs out of 5 hours

D.Scalibility study of Apriori alforithm

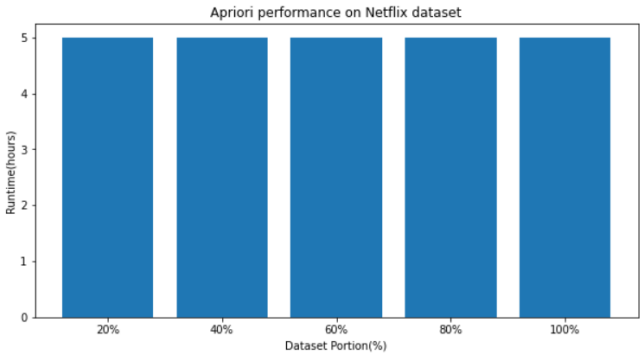

Figure 1. performance of Apriori on Netflix dataset

Since 20% of Netflix dataset already run out of time, so running time of different portion are 5+ hour.

II. WEEK 2-3: PCY ALGORITHM

A. Introduction of PCY Algorithm

PCY algorithm is a 2-pass algorithm for finding frequent pairs.

Pass 1: Count occurrences of individual items in the dataset. Additionally, maintain a hash table with as many buckets as fit in memory.

Between Passes: Replace the buckets with bit map/bit vector to save more space

Pass 2: Count pairs that meet the conditions:

1. Both items in pairs are frequent items
2. Pairs that hashed to frequent buckets

B. Implementation of PCY Algorithm

I implemented the PCY algorithm in the following steps:

1. Read dataset from file
2. Generate the hash function: (item1 + item2) % number of baskets
3. Generate the min_support threshold
4. Count the number of occurrences for each item in the dataset. This is for finding C1. In addition to counting individual items, also hash pairs in each basket to buckets
5. Filter items that occurrences are below threshold and store the frequent items. This is for finding L1
6. Generate bitmap based on buckets, if a bucket satisfied threshold, then assign 1 in the bitmap, otherwise assign 0 in the bitmap
7. Use L1 to generate unique pairs in C2, this is for avoiding checking for frequent items in Pass 2

8. Filter pairs in C2 by checking if the related bitmap value is 1 or not
9. Count the number of occurrences of the filtered pairs and store those pair that satisfied the threshold. This is for finding L2

C. *Result of PCY Algorithm*

1.The last column of the table represents the number of frequent pairs

2.The Netflix dataset gives out of memory error so there is no table built based on the Netflix dataset

Retail dataset

Time (s)	Threshold (%)	Frequent pairs
80.97 s	1%	58
14.03 s	2%	22

Netflix dataset: give out of memory error

D. *Performance Comparison Between Apriori and PCY*

The following graph shows runtime comparison between Apriori and PCY algorithm on Retail dataset. Since the Netflix dataset runs out of time limit for Apriori Algorithm and gives out of memory error for PCY algorithm, so the comparison is based on the result from Retail dataset.

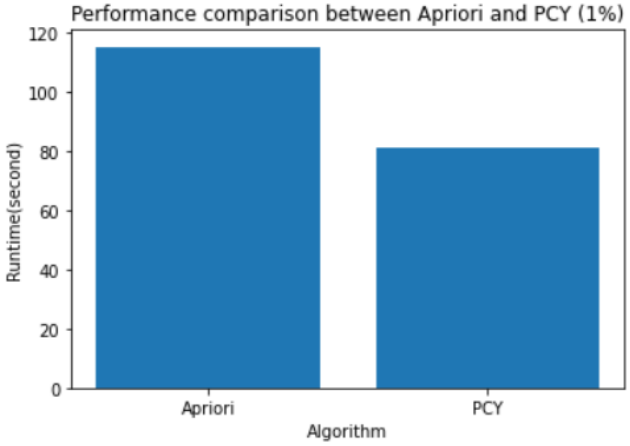

Figure 2 performance comparison between Apriori and PCY with 1% threshold

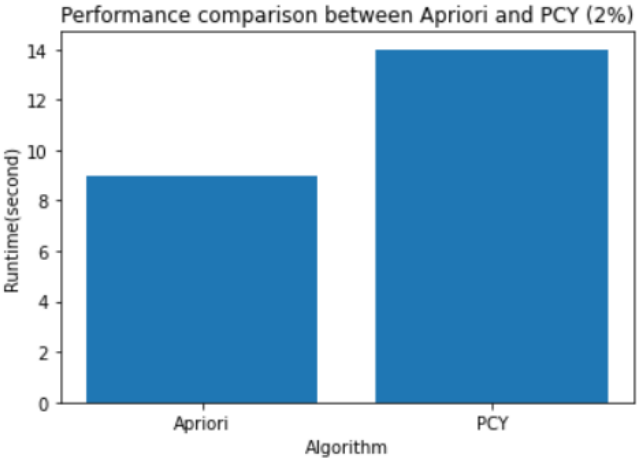

Figure 3 performance comparison between Apriori and PCY with 2% threshold

As we can see from the graph, PCY performs better when the threshold is 1% but Apriori performs better when the threshold is 2%. When the threshold is 1%, since we hash the pairs and counting individual items at pass 1 at the same time, and we erase most of the candidate pairs in the Pass 2 this really speeds up the algorithm comparing to Apriori. But when the threshold is 2% the hash table does not eliminate 2/3 candidate pairs, and hash table is stored in triple, so PCY does not beat Apriori this time.[1]

III. Week 4: Random Sampling and SON Algorithm

A. *Introduction of Random Sampling and SON Algorithm*

Random Sampling: Take a random sample of the dataset and run an in memory algorithm to get the result. Verify the result based on the original dataset[1]

SON: Repeatedly read small subsets of the orginal dataset and run in memory algorithm to find frequent pairs of each subset. In Pass 2, count the candidate pairs and find frequent pairs based on the original dataset

B. *Implementation of Random Sampling and SON Algorithm*

Random Sampling:

1. Read dataset
2. Randomly pick a sample portion of dataset (mine is 25%)
3. Use a new threshold to match the sample portion (25% of original threshold)
4. Run PCY algorithm based on the sample dataset
5. Get frequent pairs of the sample dataset
6. Verify those pairs based on the original dataset

SON:

1. read dataset
2. Slice the dataset into equal size subsets (mine sliced into 4 parts)
3. Run PCY algorithm on each subset
4. Find frequent pairs in each subset
5. Combine the frequent pairs from each subset
6. Verify all those frequent pairs using original dataset

C. *Results of Random Sampling and SON Algorithm*

Random Sampling:

1. For Netflix dataset, it runs out of 5 hours
2. The last 3 columns of the table represent the number of pairs

Retail dataset

Time(s)	Threshold(%)	Frequent pairs from subsets	Frequent pairs	False positive
30.62s	1%	57	55	2
5s	2%	22	22	0

Netflix dataset: run out of 5 hours

SON:

1. For Netflix dataset, it runs out of memory
2. The last column of the table, represent the number of frequent pairs

Retail dataset

Time(s)	Threshold(%)	Frequent pairs
418.03s	1%	50
34.76s	2%	19

Netflix dataset: Runs out of memory

D. Analysis of Random Sampling and SON Algorithm

1. Choice of sample portion for Random Sampling:
I choose to pick 25% of random baskets from the dataset, I think 25% is good enough to cover most of the truly frequent pairs in the sample dataset

2. False positive analysis for Random Sampling:
More "frequent pairs" are generated from the sample dataset due to the small sample threshold for the sample dataset. Since a larger value of sample threshold can get less false positive,we can raise the sample dataset size or increase the min_support threshold for example 1% to 2% to get less false positives.

3. Comparison of Efficiency for different algorithms
The comparison is based on the retail dataset since netflix dataset gives problems like run out of time,

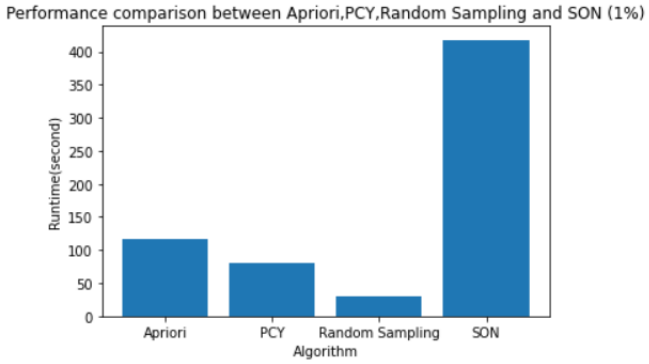

Figure 4 performance comparison between Apriori, PCY, Random Sampling and SON with 1% threshold

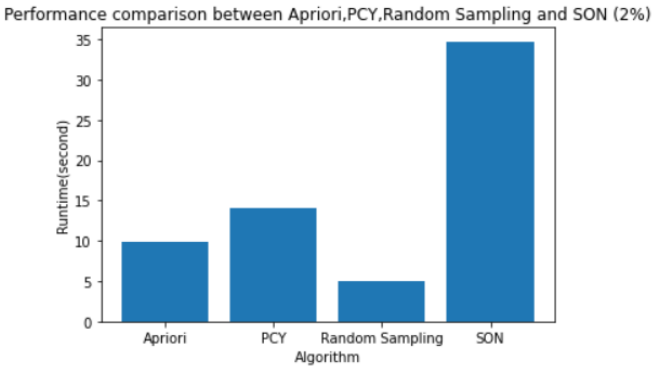

Figure 5 performance comparison between Apriori, PCY, Random Sampling and SON with 2% threshold

As we can see, the random sampling method always have the most efficiency on running time comparing to other algorithms.

IV. WEEK 5: MULTISTAGE ALGORITHM

A. Introduction of Multistage algorithm

Multistage algorithm is a refinement of PCY algorithm. After Pass 1 of PCY, rehash those qualified pairs to a new hash table with a different hash function

B. Implementation of Multistage algorithm

I implemented the multistage algorithm in the following steps:

1. Read the dataset
2. Set threshold
3. Set 2 different hash functions
4. Count individual items in the dataset and hash pairs from each basket using the first hash function to the first hash table
5. Filter items that occurrences are below threshold and store the frequent items. This is for finding L1
6. Prune the dataset by using the frequent items from L1, if a basket does not contain any frequent item, remove it from the dataset
7. Generate pairs based on the frequent items in L1 and generate the first bit map
8. Rehash the qualified pairs based on the first bit map to another hash table using the second hash function
9. Generate the second bit map
10. Use the generated pairs in step 7 and filter it using the first bit map and the second bitmap, verify the remaining pairs with the original dataset

C. Results of Multistage Algorithm

1.The last column of the table represents the number of frequent pairs

2.The Netflix dataset gives out of memory error so there is no table built based on the Netflix dataset

Retail dataset

Time (s)	Threshold (%)	Frequent Pairs
211.57	1%	58
25.86	2%	22

V. Problems and Optimization

Problem:
The most obvious problem during the project is my algorithm can not handle the netflix dataset, I think the reason is not just the number of baskets in the Netflix dataset is for more than the retail dataset, but the average length of baskets in Netflix dataset is much larger than baskets in Retail dataset

Optimization:
Use Numpy Array to store data: In all of my algorithms, I stored the dataset, L1, C2 using list in python, this consume more space than numpy array and in numpy tasks are segmented into small parts to process in parallel.

Prune the dataset by using frequent items(L1): Not all of my algorithm, prune the dataset based on frequent items. I think when the frequent items are genrerated, we can use it to prune the dataset used for coming operations. If a basket does not contain any frequent item, we can remove that basket from the dataset

VI. Important note

1.All my algorithm is written using Jupiter Notebook
2. I can not get subset of frequent pairs from Netflix dataset for PCY algorithms, Random Sampling Algorithm, SON algorithm and Multistage algorithm since they are all based on the PCY algorithm and when I run Netflix dataset on it, it gives out of memory error
3.The result pair in the txt file is in format (item1,item2)

References

[1]Week2 slides Finding association rule: https://eclass.yorku.ca/pluginfile.php/4215095/mod_resource/content/6/BDA_06-assocrules.pdf